```
# # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # #
#
LocationToBinary <- function(n){
  binary.vector <- rev(as.numeric(intToBits(n)))
  first.1 <- (which(binary.vector == 1)[1])
  binary.vector <- binary.vector[-(1:(first.1-1))]
  return(binary.vector)
}
#
# # # # # # # # # # # # # # # # # # # # # # #
#
# system = 1 Calkin-Wilf
# system = 2 driB
# system = 3 Yu-Ting
# system = 4 Yurramendi-1
#
BinaryToCF.1.1 <- function(binary.vector){
  # filter   <- rep(c(1,1), 16)[1:length(binary.vector)]
  # nbin <- as.numeric(binary.vector == filter)
  nbin <- binary.vector
  dnbin <- c(nbin[-length(nbin)] - nbin[-1] , 1)
  cuts <- which(dnbin != 0)
  runlengths <- c(cuts[1], cuts[-1] - cuts[-length(cuts)])
  cf <- rev(runlengths)
  if(binary.vector[length(binary.vector)] == 0) cf <- c(0, cf)
  return(cf)
}
#
BinaryToCF.1.2 <- function(binary.vector){
  filter   <- rep(c(1,0), 16)[1:length(binary.vector)]
  nbin <- as.numeric(binary.vector == filter)
  dnbin <- c(nbin[-length(nbin)] - nbin[-1] , 1)
  cuts <- which(dnbin != 0)
  runlengths <- c(cuts[1], cuts[-1] - cuts[-length(cuts)])
  cf <- rev(runlengths)
  if(binary.vector[length(binary.vector)] == 0) cf <- c(0, cf)
  return(cf)
}
#
BinaryToCF.1.3 <- function(binary.vector){
  nbin <- binary.vector
  ones <- which(nbin == 1)
  if(nbin[length(nbin)] == 0) ones  <- c(ones, length(nbin))
  dones <- ones[-1] - ones[-length(ones)]
  cf <- rev(c(ones[1], dones))
  if(nbin[length(nbin)] == 0) cf <- c(0, cf)
  return(cf)
}
#
BinaryToCF.1.4 <- function(binary.vector){
  nbin <- binary.vector
  zeros <- c(which(nbin == 0), length(nbin))
  dzeros <- zeros[-1] - zeros[-length(zeros)]
  cf <- rev(c(zeros[1], dzeros))
  return(cf)
}
#
#
CFToFraction <- function(cf){
  r <- cf[1]
  if(length(cf) > 1){
    r <- 1/cf[length(cf)]
    for(i in 1:(length(cf)-1)) r[i+1] <- 1/(cf[length(cf)-i] + r[i])
    r[length(cf)] <- cf[1] + r[length(cf)-1]
  }
  return(fractions(r[length(r)]))
}
#
# # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # #
#
LocationToFraction.CF.1 <- function(system, n){
#
#   system = 1  Calkin-Wilf
#   system = 2  driB
#   system = 3  Yu-Ting
#   system = 4  Yurramendi-1
#
# Location --> Binary --> Continued Fraction --> Fraction
#
# Location --> Binary
#
  binary.vector <- LocationToBinary(n)
#
#             Binary --> Continued Fraction
#
  if(system == 1)  cf <- BinaryToCF.1.1(binary.vector)
  if(system == 2)  cf <- BinaryToCF.1.2(binary.vector)
  if(system == 3)  cf <- BinaryToCF.1.3(binary.vector)
  if(system == 5)  cf <- BinaryToCF.1.5(binary.vector)
  if(system == 4)  cf <- BinaryToCF.1.4(binary.vector)
#
#                        Continued Fraction --> Fraction
#
  fraction <- CFToFraction(cf)
#
  return(fraction)
}
#
# # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # #
#
# Examples: Compute fraction of the 47th in all four systems
#
library(MASS)
#
LocationToFraction.CF.1(1, 47) #  9/2
LocationToFraction.CF.1(2, 47) # 11/7
LocationToFraction.CF.1(3, 47) # 11/7
LocationToFraction.CF.1(4, 47) #  9/2
#
# # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # #
```